

Einari Vaaras

AUTOMATIC CLASSIFICATION OF INFANT- AND ADULT-DIRECTED SPEECH FROM ACOUSTIC SPEECH SIGNALS

Faculty of Information Technology and Communication Sciences
Bachelor's Thesis
May 2019

ABSTRACT

Einari Vaaras: Automatic Classification of Infant- and Adult-directed Speech from Acoustic Speech Signals

Bachelor's Thesis, 31 pages, 2 appendix pages

Tampere University

Bachelor's Degree Programme in Electrical Engineering

May 2019

Major: Signal Processing and Machine Learning

Examiner: Okko Räsänen, D. Sc. (Tech.)

Paralinguistic speech processing (PSP) is a field of audio processing where the focus of the analysis is beyond the literal message. One potential task in the area of PSP is the classification of samples into infant-directed speech (IDS) and adult-directed speech (ADS). In the present study, a system which classifies samples into IDS/ADS as accurately as possible was examined by experimenting with different classifiers used in other fields of PSP, and by testing different sets of manually defined features. The classification results showed that the best set of features was a set which included all speech-relevant features extracted in this study except spectrogram. Additionally, support vector machines (SVMs) performed the best of the individual classifiers used in the study, while an ensemble classifier outperformed all individual classifiers. These results were in line with the previous IDS/ADS classification studies in the field.

Keywords: classification, speaking style, infant-directed speech, adult-directed speech, motherese, speech processing, acoustic signal

The originality of this thesis has been checked using the Turnitin OriginalityCheck service.

PREFACE

This thesis marks the first major milestone in my studies at the present Tampere University and the former Tampere University of Technology. I would like to thank Prof. Okko Räsänen, the examiner of this thesis, for all the guidance he has given me throughout this project and for the vast amount of things I have learned in the field of speech processing during the past few months. I would also like to thank Prof. Joni Kämäräinen for indirectly guiding me towards the path of studying signal processing and machine learning. Something about signal processing fascinated me when I started my university studies, and the only point of contact with me and the subject was that I knew someone working in the field.

Additionally, I would like to express my gratitude to everyone who has been involved in this project directly or indirectly, including all my friends who have supported me along the way. Most sincerely I would like to thank Siru Peltoniemi for providing me with the energy and motivation to work on this thesis each and every day. Without her, this thesis would not be at the level where it is now.

Tampere, 03.05.2019

Einari Vaaras

CONTENTS

1. INTRODUCTION	1
2. LITERATURE REVIEW	3
3. METHODS	7
3.1 Digital Representation of an Acoustic Signal	7
3.2 Feature Extraction	8
3.2.1 Spectrogram	8
3.2.2 Complex Cepstrum	10
3.2.3 Mel-frequency Cepstral Coefficients	11
3.2.4 Fundamental Frequency	11
3.2.5 Short-time Zero-crossing Rate	12
3.2.6 Short-time Energy	13
3.3 Classifiers	14
3.3.1 Support Vector Machine (SVM)	14
3.3.2 Multilayer Perceptron (MLP)	17
3.3.3 Random Forests	18
3.3.4 k -Nearest Neighbors (k -NN)	20
4. EXPERIMENTAL SETUP	22
4.1 Dataset	22
4.2 Setup	22
5. RESULTS	25
6. CONCLUSION	26
REFERENCES	28
APPENDIX A: MFCCS, DELTAS AND DELTA-DELTAS OF THE TEST SIGNAL ..	32

LIST OF FIGURES

Figure 1. A block diagram of a simple PSP system.	3
Figure 2. The digital representation of a signal waveform.	7
Figure 3. The spectrogram of the example signal in Figure 2.	9
Figure 4. The estimate of the PSD of the example signal in Figure 2.	9
Figure 5. The complex cepstrum of a voiced segment.	10
Figure 6. The estimate of the fundamental frequency.....	12
Figure 7. The signal waveform and the STZCR of the signal.....	13
Figure 8. The signal waveform and the STE of the signal.	14
Figure 9. The use of a kernel function.	16
Figure 10. An example of a simple decision tree.	19
Figure 11. The MFCCs of the example signal in Figure 2.....	32
Figure 12. The deltas of the example signal in Figure 2.	32
Figure 13. The delta-deltas of the example signal in Figure 2.	33

LIST OF SYMBOLS AND ABBREVIATIONS

ACLEW	Analyzing Child Language Experiences Around the World
ADS	Adult-directed speech
BoAW	Bag-of-audio-words
CNN	Convolutional neural network
DFT	Discrete Fourier transform
DNN	Deep neural network
DTFT	Discrete-time Fourier transform
f_0	Fundamental frequency
FFT	Fast Fourier transform
FNN	Feed-forward neural network
IDFT	Inverse discrete Fourier transform
IDS	Infant-directed speech
IDTFT	Inverse discrete-time Fourier transform
k -NN	k -nearest neighbors
MFCCs	Mel-frequency cepstral coefficients
MLP	Multi-layer perceptron
NN	Nearest neighbor
PSD	Power spectral density
PSP	Paralinguistic speech processing
ReLU	Rectified linear unit
SRH	Summation of residual harmonics
STE	Short-time energy
STFT	Short-time Fourier transform
STZCR	Short-time zero-crossing rate
SVM	Support vector machine
UAR	Unweighted average recall
\times	Cartesian product
\cdot	Dot product
$\ \cdot \ $	Euclidean norm
$[a, b]$	A closed value range from a to b
$\arg()$	The argument of a complex number
j	Imaginary unit
$M \times N$	A matrix of size $M \times N$
\mathbb{R}	Set of real numbers
\mathbb{Z}	Set of integers

1. INTRODUCTION

Digital speech processing typically deals with the formal structure of the language, which is called the linguistic content. The linguistic content is the aspect of speech which consists of phonemes, words and sentences (Rabiner & Schafer, 2011). However, speech consists of much more information than what is being said in how it is said, including the personality, mood, emotion, social background, and health of the speaker (Schuller & Batliner, 2013, pp. 3-20; Schuller et al., 2013). As an example, one can typically hear from a recording whether the person speaking is a child or an adult, whether the person is a male or a female, what the person's dialect is, and whether the person is intoxicated or not. The digital analysis of speech beyond the literal message is called computational paralinguistics, in which the term 'computational' refers to something being done by a computer, and the term 'paralinguistics' means 'alongside linguistics': it is concerned with the way something is said rather than what is being said. Computational paralinguistics, also known as paralinguistic speech processing (PSP), is a rather new field of study, since a little over 20 years ago the field did not exist (Schuller & Batliner, 2013, pp. 3-20; Schuller et al., 2013).

One subcategory of PSP is the classification between speaking styles, where the discrimination of infant-directed speech (IDS) from adult-directed speech (ADS) is one topic of interest. IDS is a speaking style typically used in interaction between infants having distinctive properties compared to ADS, including reduced speed (Eaves et al., 2016), higher fundamental frequency (f_0), higher frequency range (Soderstrom, 2007), shorter utterances, and intonational exaggeration (Fernald et al., 1989). There is also language-specific variation between the properties of IDS speech which indicates that universal generalization of IDS properties is not possible from the empirical observations from an individual culture (Fernald et al., 1989; Soderstrom, 2007).

The role of IDS and the roles of the different properties of IDS in the language development of an infant have been discussed and hypothesized in a plethora of studies (e.g. Fernald et al., 1989; Soderstrom, 2007; Rowe, 2008; McMurray et al., 2013; Spinelli et al., 2017). It has been shown that especially newborn infants prefer IDS over ADS (Fernald, 1985; Cooper & Aslin, 1990; Pegg et al., 1992), but there has been very little study on what the properties of IDS are that are linked with the higher attentional capture or

with the cognitive and emotional development of an infant (e.g. Saint-Georges et al., 2013; Butler et al., 2014; Eaves et al., 2016; Räsänen et al., 2018).

The automatic differentiation between IDS and ADS is interesting from the language research point of view, since an automatic system capable of classifying IDS and ADS would help vastly in the study of language development of children by increasing the amount of data available, replacing the time-consuming manual annotation of audio recordings. For instance, one ongoing project focusing on the study of children's language development is called Analyzing Child Language Experiences Around the World (ACLEW; <https://sites.google.com/view/aclewdid>; 2017-2020) which aims to make large-scale collections of audio recordings of children's everyday lives accessible to the research community by developing open-source tools for the automated analysis of real-world language recordings, including automatic classification of speech into IDS and ADS.

The purpose of this thesis is to contribute to the work in ACLEW by creating a system which classifies speech samples into IDS or ADS as accurately as possible by studying different classification and feature extraction methods in the task. Since the classification between IDS and ADS is a subcategory of PSP, the methods used in other fields of PSP can be taken advantage of. Methods previously used in PSP and IDS/ADS classification are discussed in Chapter 2. Chapter 3 gives an overview of the extracted features and the classification methods used in the present study, followed by a detailed description of the dataset and the used experimental setup in Chapter 4. Finally, the results of the study are presented in Chapter 5, and conclusions are drawn in Chapter 6.

2. LITERATURE REVIEW

As mentioned in the previous chapter, the classification between IDS and ADS can be considered as an area of PSP. A typical PSP system consists of training a classifier with supervised learning using features extracted from speech data (Schuller & Batliner, 2013, pp. 179–184), and using the classifier to predict the class of an unknown sample. Supervised learning consists of labeling data, training a system with features extracted from the dataset, and then validating the system using a validation set that is separate from the dataset that is used for training the system. A block diagram of a simple PSP system is depicted in Figure 1.

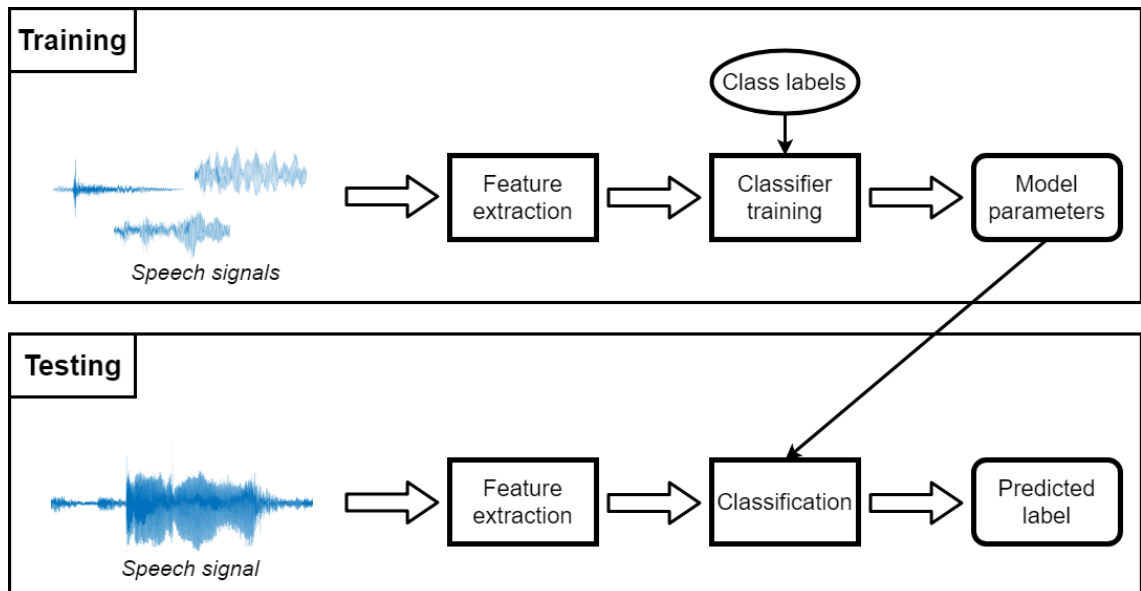


Figure 1. A block diagram of a simple PSP system.

One key role for classification in PSP is played by feature extraction (the second block in Figure 1) (Schuller & Batliner, 2013) which typically begins with short-time features extracted from 10–30 ms windows of the digital speech signal, often using smooth windowing functions (e.g. Hamming). The signal is segmented into these smaller windows to represent the frequency content of the signal as it varies over time. To avoid data loss using a smooth window, the adjacent windowed audio frames are overlapped, typically with time shifts of 10 ms (Schuller & Batliner, 2013, pp. 179–189). After windowing, features are extracted from the windowed segments. These features include the mel-frequency cepstral coefficients (MFCCs), the short-time zero-crossing rate (STZCR), f_0 and the FFT spectrum of the signal inside the analysis window (Schuller & Batliner, 2013, pp. 179–184, 289–294).

However, by only focusing on the frame-level information obtained from the feature extraction, valuable paralinguistic information is lost. Typically, paralinguistic classification is performed at the level of utterances, i.e. continuous speech ending with a clear pause, which results in the need of using the extracted featural information across multiple frames (Schuller & Batliner, 2013, pp. 230–234; Schuller et al., 2013). This leads to the requirement of obtaining a fixed-length feature vector from samples with varying lengths. One common method of obtaining a feature vector with a fixed length from varying inputs is to apply functionals to the time series of the extracted features. These functionals can include e.g. the mean, variance, skewness, kurtosis and the minimum and the maximum of the time series corresponding to the feature of interest (Schuller & Batliner, 2013, pp. 230–234).

The selection of features in PSP depends on the domain knowledge of the subject at hand and the amount of training data available. With much knowledge of the examined topic, less data is needed when using manually tailored features that minimize unwanted variability, but as a downside relevant information might be lost when using a small data set and less features. With a large dataset, representation learning can be used to train a model directly from the acoustic waveforms. This, on the other hand, requires a larger amount of data and more extensive computation. Since PSP tasks usually involve relatively sparse datasets, a common solution to outcome the need for manually tailored features or a large dataset is to extract a vast pool of different features by a feature extraction toolkit (for example openSMILE¹; open-source Speech and Music Interpretation by Largespace Extraction), and then to test which features contribute best to the classification accuracy. Another common solution is to use the full set or a large set of features in conjunction with a classifier that is robust to non-significant features (Batliner et al., 2010; Bengio et al., 2013; Schuller & Batliner, 2013, pp. 179–184, 230–280, 289–303).

Traditionally support vector machines (SVMs), random forests, and more recently neural networks like multilayer perceptrons (MLPs) and convolutional neural networks (CNNs) have been used in PSP classification problems, of which many have been discussed in annual challenges held at Interspeech conferences (<http://compare.openaudio.eu/>). There are also occasional instances of studies making use of other classifiers like k -nearest neighbors (k -NN) and Gaussian mixture models (GMMs). Of the classifiers listed above, SVMs are the most commonly used ones in PSP classification. However, the

¹ For further reading, see e.g. (Eyben et al., 2013).

classification results have varied largely depending on the task of the PSP classification (Batliner et al., 2010; Schuller et al., 2019).

PSP related to IDS has been examined in multiple studies before with some focusing on the special properties of IDS and others on the classification between IDS and ADS with varying results. Shami and Verhelst (2007) studied classification of expressiveness in speech within and across corpora with IDS and ADS databases using SVM, k -NN and random forest classifiers. Batliner et al. (2008) investigated the acoustic features which contribute best to the classification of intimacy in speech using IDS, ADS and pet-directed speech (PDS) by means of SVMs and random forests. Mahdhaoui and Chetani (2009) and Mahdhaoui et al. (2010) studied different IDS/ADS classification systems with multiple feature sets using SVMs, MLPs, k -NNs, GMMs and a combination of classifiers with the combined classifier yielding best results, resulting in a classification accuracy of 86%. However, their data was limited in size and poorly described, making it unclear how well their findings generalize beyond their experiments. Schuster et al. (2014) in turn, presented an automated end-to-end system that segments the speech of an audio file and then classifies each segment into IDS and ADS. The classification of this system was tested with SVM, decision tree, random forest, logistic regression and Naïve Bayes classifiers with SVMs performing the best with a classification accuracy of 83%. In this case the dataset used was larger and better depicted than the previously mentioned result, but, on the other hand, the reported performance level may be somewhat optimistic since the recordings were only taken from female speakers in laboratory conditions.

The aforementioned studies might provide somewhat optimistic classification performance levels in the IDS/ADS classification task, since the datasets used have been constrained in some way. A more representative result of real-world performance was obtained from INTERSPEECH 2017 computational paralinguistics challenge (Schuller et al., 2017), where IDS/ADS classification was one of the three tasks in the challenge. In the IDS/ADS classification task, the dataset used was from real-world child language recordings. In the challenge baseline system, varying sets of features were extracted using CNNs, openSMILE and a bag-of-audio-words (BoAW), and then subsequent recurrent networks and SVMs were used for classification along with combined classifiers using majority voting and the sum of confidence values. The classification results varied in the range of 59.1% to 70.2%, where the accuracies were measured using unweighted average recall (UAR). A combined classifier using the sum of confidence values was found to be the best performing one.

In the present study a more systematic approach is taken in order to create a system which classifies audio samples into IDS and ADS. In this thesis, feature extraction is

studied by manually defining different feature sets, and testing which features contribute best to the classification accuracy. In addition, different classifiers are examined with the purpose of creating the best performing classifier without the prior knowledge of which classifier works the best in action. As with PSP typically, the main challenge of creating an IDS/ADS classifier is to avoid overfitting the model to the data, since the dataset used in the study is relatively small which is known to increase the risk of overfitting.

3. METHODS

In this chapter, the basic supervised classification pipeline (as depicted in Figure 1) is further examined by applying it to the present study. Section 3.1 discusses the first part of the block diagram involved in the representation of speech signals for digital processing, while Section 3.2 advances into the next block of feature extraction and depicts the different features that are extracted from the dataset used in this study. Finally, Section 3.3 represents the parts of the block diagram involved with classification by examining the different classifiers used in the present study. In this chapter, the different speech signal representations are demonstrated using a short 600-ms IDS utterance of a female saying “*there you go*”.

3.1 Digital Representation of an Acoustic Signal

According to the *IEEE Standard Dictionary of Electrical and Electronics Terms*, a signal is a physical representation which conveys data from one point to another (Jay, 1984). The word acoustic, in turn, is by one definition something related to sound or the science of sound waves (Beranek & Mellow, 2012, p. 10). Therefore an acoustic signal is a sound signal carrying information of some kind.

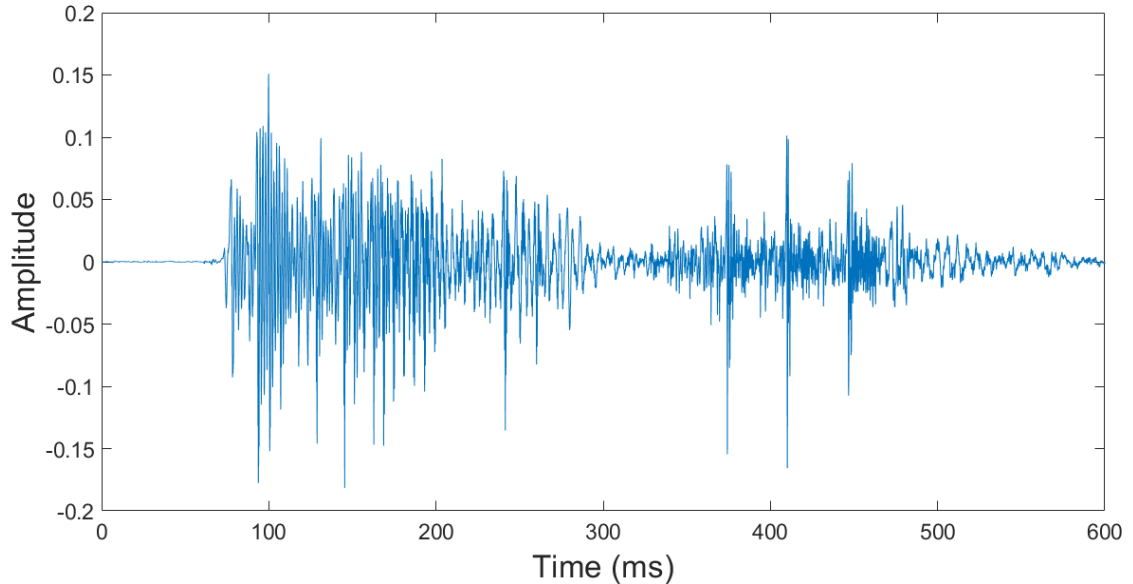


Figure 2. The digital representation of a 600-ms signal waveform, in which a woman says “*there you go*”.

A physical (analog) signal is observed as a continuous-time function $s(t)$. In order to be processed digitally, it has to be sampled into a sequence of discrete-time values $x(rT)$,

where $r \in \mathbb{Z}$ and T is the time period between two samples such that $t = rT$. With speech signals, this quantization is typically done using a sampling frequency of 16 kHz in order to preserve the frequency range of $[0, 8]$ kHz, which is a common frequency band used for speech. In addition, the values of the continuous-time function $s(t)$ must be quantized before the signal can be represented in digital format. Typically, a 16-bit representation, which is the same number of bits used for example in e.g. CDs and WAV files, is used with digitalized speech. The 16-bit representation results in the dynamic range of sound pressure being from 0 dB up to approximately 96 dB if uniformly spaced quantization is used (Tohyama & Koike, 1998, pp. 4–33). An example of a digitalized 600-ms IDS utterance in which a woman says “*there you go*” is shown in Figure 2.

3.2 Feature Extraction

This section provides an overview of the different features extracted in this study. Feature extraction plays an integral part in PSP, and it is the second part of the block diagram in Figure 2.

3.2.1 Spectrogram

A spectrogram is a time-frequency intensity display of a short-time spectrum used to visually represent the frequency content of a signal varying with time (Oppenheim, 1970). The spectrogram is widely used in audio processing as an analysis tool for numerous applications (Gold et al., 2011). To obtain the spectrogram of a digital signal, the signal is windowed, and the squared magnitude of the discrete STFT of the windowed signal is taken, in other words

$$\text{spectrogram}(t, \omega) = |\text{STFT}(t, \omega)|^2 = \left| \sum_{n=0}^{N-1} x(n)w(t-n)e^{-j\omega n} \right|^2, \quad (3.1)$$

where t is the time instant of analyzation, ω is the analysis frequency, $x(n)$ is the time domain signal, $w(n)$ is the windowing function, and N is the window length (Rabiner & Schafer, 2007, pp. 42–46).

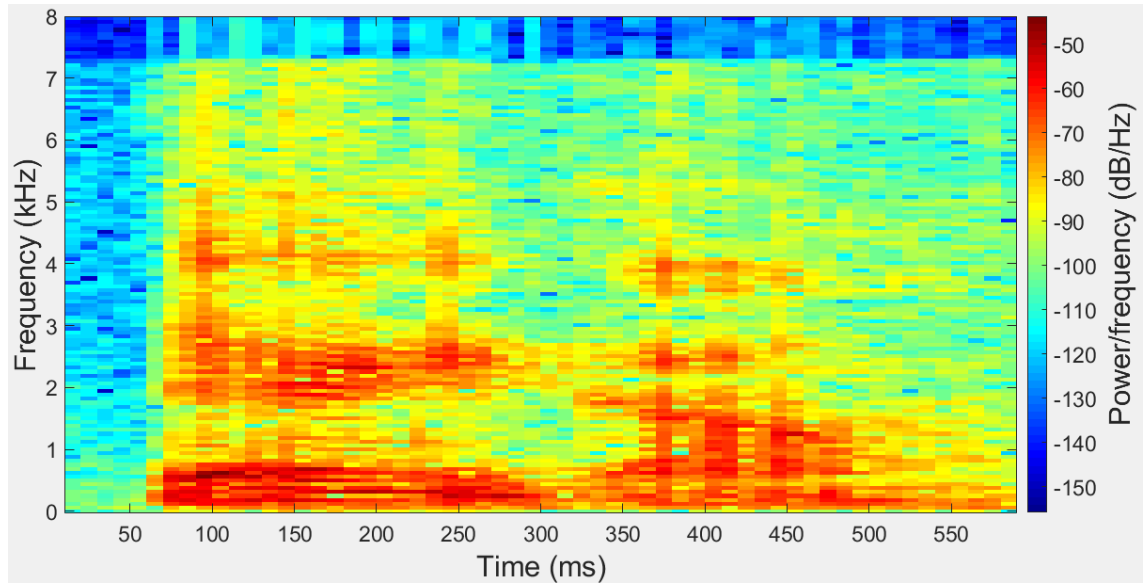


Figure 3. The spectrogram of the example signal in Figure 2.

Windowing means segmenting the signal in time domain into smaller overlapping sections that are typically the length of tens of milliseconds, and softening the edges of the windows with a windowing function, typically a Hanning or Hamming window. Determining the window length is always balancing between the resolutions of the time and the frequency axis, as a short analysis window results in a good time resolution and a poor frequency resolution, and vice versa for a long analysis window (Rabiner & Schafer, 2007, pp. 46–49). An example image of a spectrogram is shown in Figure 3 which depicts the spectrogram of the example signal in Figure 2 with 128 frequency bins. The spectrogram is calculated using a 30-ms window with a time shift of 10 ms.

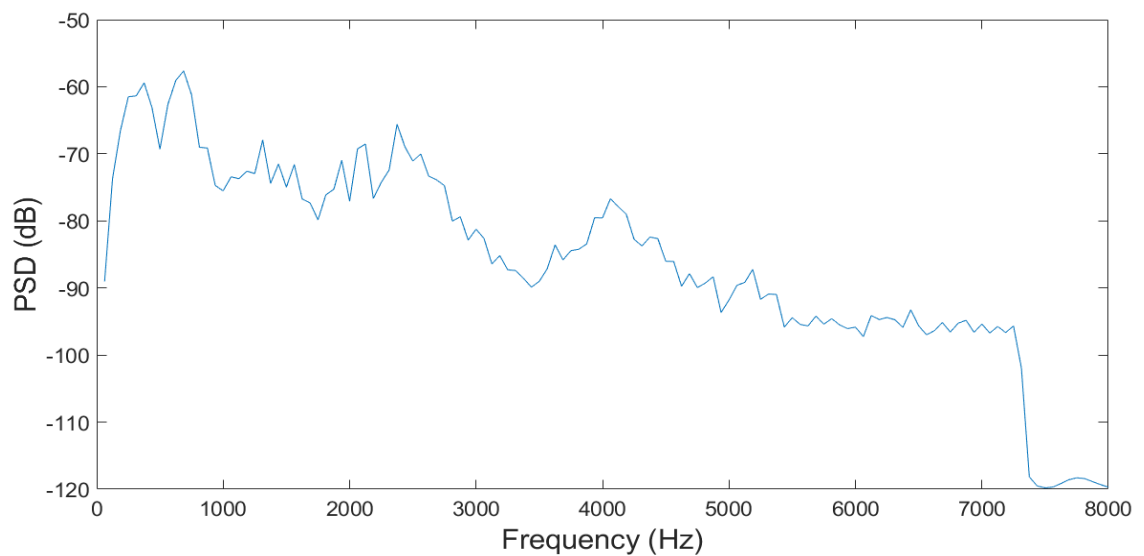


Figure 4. The estimate of the PSD of the example signal in Figure 2.

The long-term time average of a spectrogram (with power spectrum) is also sometimes referred to as a periodogram (Kay, 1988). The periodogram is the estimate of the average power spectral density (PSD) of a signal. In Figure 4, the estimate of the PSD of the example signal in Figure 2 is demonstrated using 128 frequency bins.

3.2.2 Complex Cepstrum

The complex cepstrum $\hat{x}(n)$ of a digital signal $x(n)$ is the IDFT of the complex logarithm of the magnitude of the DFT, which Rabiner and Schafer (2007, pp. 54–59) defined as

$$\hat{x}(n) = \frac{1}{N} \sum_{k=0}^{N-1} \hat{X}(k) e^{jn \frac{2\pi k}{N}}, \quad (3.2)$$

where N is the number of samples in $x(n)$ and $\hat{X}(k)$ is the complex logarithm of the magnitude of the DFT, which is defined as

$$\hat{X}(k) = \log \left| \sum_{n=0}^{N-1} x(n) e^{-jn \frac{2\pi k}{N}} \right| + j \cdot \arg \left(\sum_{n=0}^{N-1} x(n) e^{-jn \frac{2\pi k}{N}} \right). \quad (3.3)$$

By using the DFT of the signal $x(n)$ instead of the continuous-time DTFT, the complex cepstrum $\hat{x}(n)$ is an approximation of the true complex cepstrum, since time-domain aliasing occurs because of sampling the log of the DTFT. This can be compensated by using a large value of N (Oppenheim et al., 1999, pp. 629–646). Figure 4 shows the complex cepstrum of a voiced 30-ms segment of the example signal in Figure 2.

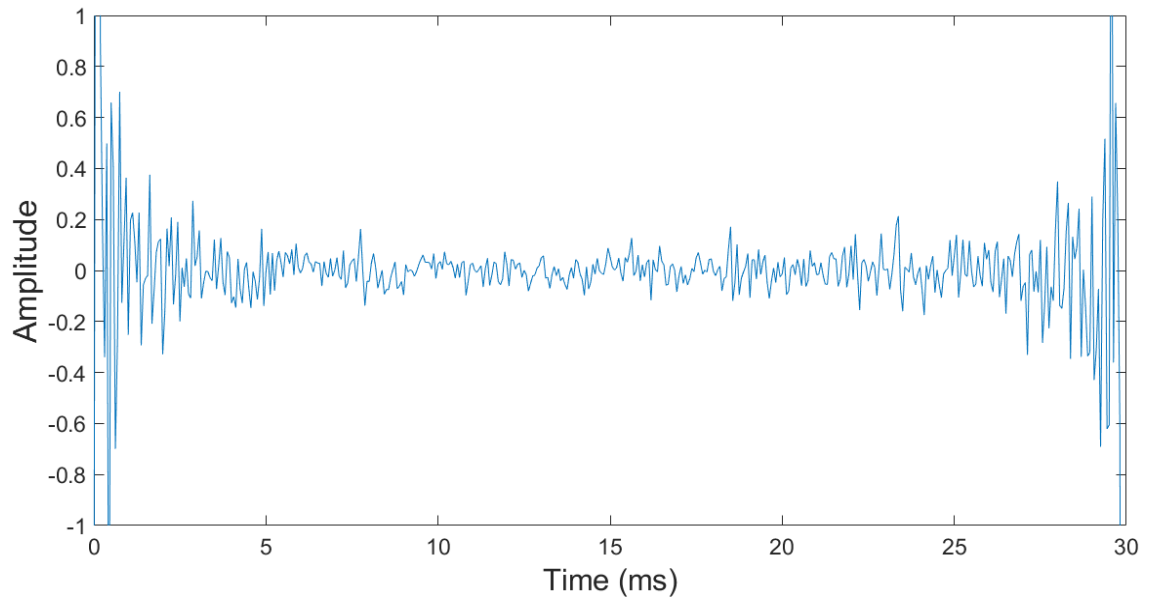


Figure 5. The complex cepstrum of a voiced segment of the signal in Figure 2.

The complex cepstrum depicts the transition rate in different spectrum bands and is widely used in speech analysis, namely in pitch detection and in estimating parameters

of a speech production model. The smallest order cepstral coefficients model the spectral properties of the vocal tract, and the higher-order cepstral coefficients represent the glottal excitation in the source-filter model² (O’Shaughnessy, 1999, pp. 173–225).

3.2.3 Mel-frequency Cepstral Coefficients

The mel-frequency cepstral coefficients (MFCCs) are a cepstrum representation that takes the human perception of sound into account. This is achieved by spacing the frequency bands of the signal on the mel scale, which is a nonlinear scale that models the way humans perceive the frequency of a sound also known as the pitch (Rabiner & Schafer, 2007, pp. 29–31, 69–71). A mel-frequency filter bank which consists of R triangular weighting functions is used for the DFT magnitude spectrum of the signal. Optionally, mel-scale bin energies can be normalized in order to get a flat mel-spectrum from a flat input Fourier spectrum. As with normal magnitude and power spectra, this representation loses the phase information which is included in the complex cepstrum, but is not typically considered important in speech processing applications. This is because typically the phase information is dependent on the audio recording setup and transmission channel, and thus does not have a significant effect on the understandability of speech. A discrete cosine transform of the logarithm of the magnitude of the filter outputs is computed for each frame to produce the MFCCs. The MFCCs $c(n)$ are defined as

$$c(n) = \frac{1}{R} \sum_{r=1}^R \log(E(r)) \cos \left[\frac{2\pi}{R} \left(r + \frac{1}{2} \right) n \right], \quad (3.4)$$

where $E(r)$ is the normalized and triangular-weighted DFT bin energy at time instant r (Rabiner & Schafer, 2007, pp. 69–71). Typically, $c(n)$ are evaluated for 13 coefficients. Additional coefficients such as the delta and the delta-delta, which indicate the first- and second-order frame-to-frame difference, are usually calculated together with the 13 $c(n)$ to obtain 39 coefficients altogether (Furui, 1986; Rabiner & Schafer, 2007, pp. 69–71). The MFCCs, deltas and delta-deltas of the example signal in Figure 2 are depicted in Appendix A.

3.2.4 Fundamental Frequency

Estimating the fundamental frequency (f_0) of voiced speech segments is used in various applications of speech processing such as speech recognition (Drugman & Alwan, 2011). During voiced speech, f_0 corresponds to the vibration frequency of the vocal folds,

² For further reading, see e.g. (Pulkki & Karjalainen, 2014, pp. 79–97).

also known as the frequency of the glottal excitation. The estimate of f_0 is an important factor in the analysis of speech since the f_0 of the glottal excitation determines the pitch of the voice that is perceived (Rabiner & Schafer, 2007, pp. 20–23, 126–128).

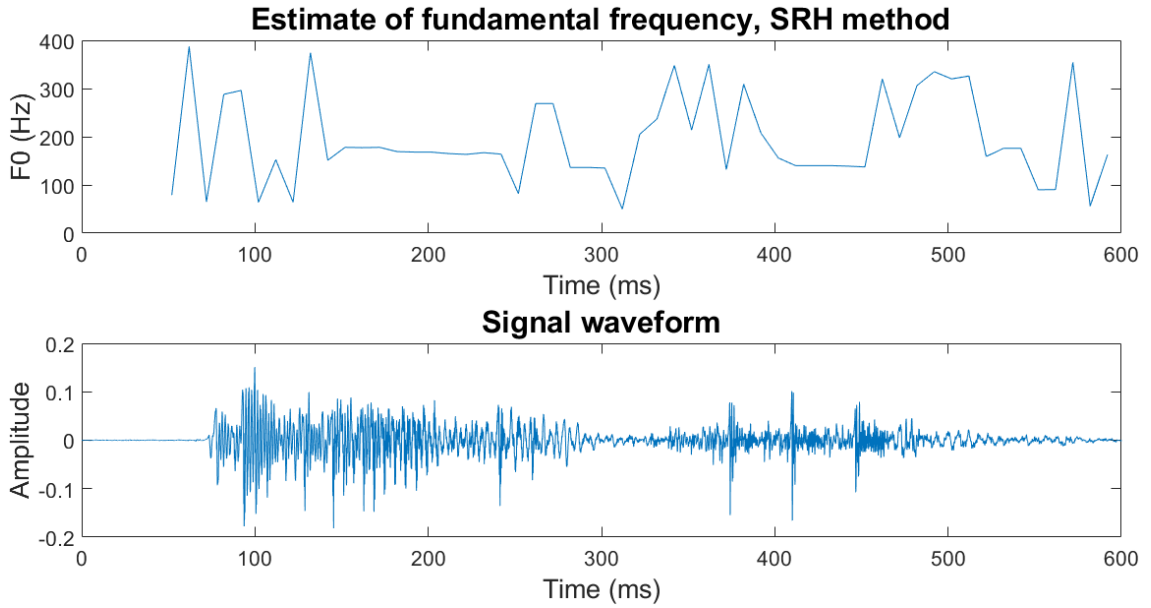


Figure 6. The estimate of the fundamental frequency using the SRH method (upper image) and the signal waveform (lower image).

The calculation method of f_0 in this study is the summation of residual harmonics (SRH) which is capable of estimating the f_0 well in noisy conditions (see Drugman & Alwan, 2011, for an overview). This is a desired property, considering the naturalness of the recordings used in the dataset of the present study. Figure 5 demonstrates the estimate of the fundamental frequency of the example signal in Figure 2 using the SRH method.

3.2.5 Short-time Zero-crossing Rate

The short-time zero-crossing rate (STZCR) is the weighted average of the number of times a signal changes sign within a time window, i.e. (Rabiner & Schafer, 2007, pp. 37–40)

$$\text{STZCR} = \sum_{n=-\infty}^{\infty} \frac{1}{2} |\text{sgn}[x(n)] - \text{sgn}[x(n-1)]| w(m-n), \quad (3.5)$$

where $x(n)$ is the time domain signal, $w(n)$ is the windowing function, m is the time index of the block of samples in the windowing function, and

$$\text{sgn}[x] = \begin{cases} 1, & x \geq 0 \\ -1, & x < 0. \end{cases} \quad (3.6)$$

STZCR acts as a simple-to-compute frequency analyzer (Rabiner & Schafer, 2007, pp. 37–40), providing valuable information about whether a section of a speech signal is voiced or unvoiced.

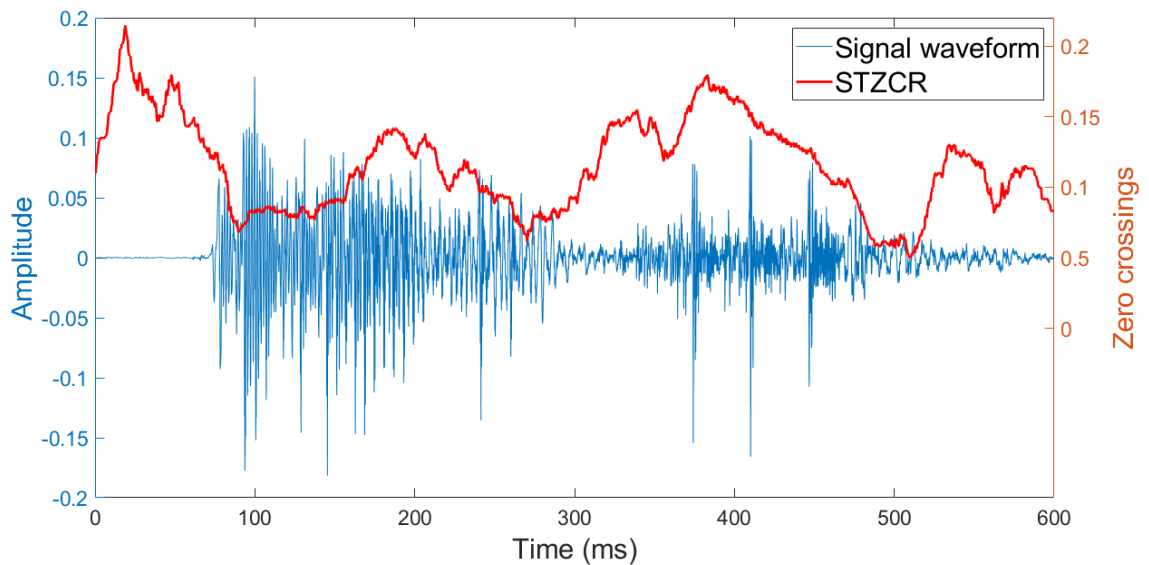


Figure 7. The signal waveform and the STZCR of the signal.

Figure 7 depicts the STZCR of the example signal in Figure 2 using a rectangular 40-ms window. The figure demonstrates that the parts of the signal with unvoiced speech have a higher STZCR, and with voiced speech the STZCR is lower.

3.2.6 Short-time Energy

The short-time energy (STE) of a signal is defined as

$$\text{STE} = \sum_{n=-\infty}^{\infty} [x(n)w(m-n)]^2, \quad (3.7)$$

where $x(n)$ is the time domain signal, $w(n)$ is the windowing function, and m is the time index of the block of samples in the windowing function (Rabiner & Schafer, 2007, pp. 37–40). The STE indicates the amplitude of the signal in the interval around time m , which results in unvoiced sections of speech having lower STE than voiced regions. This makes the STE a useful tool in detecting voiced and unvoiced sections of utterances.

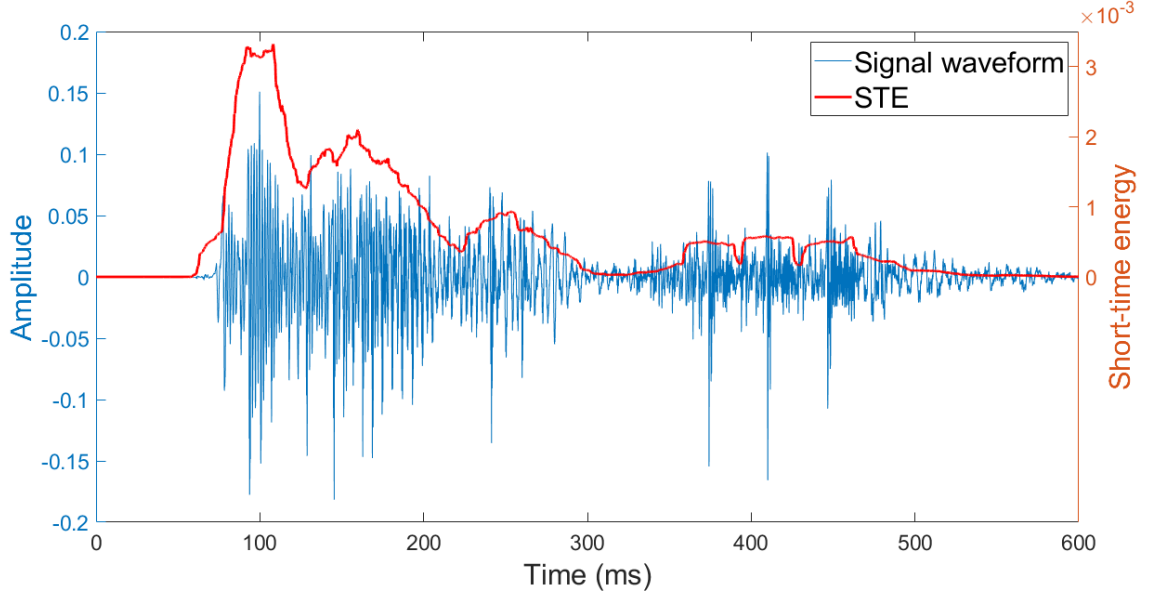


Figure 8. *The signal waveform and the STE of the signal.*

Figure 8 depicts the STE of the example signal in Figure 2 using a rectangular 40-ms window. It can be seen from the image that the unvoiced parts of the signal have a lower STE and the voiced parts have a higher STE.

3.3 Classifiers

This section provides an overview of the different classifiers used in this study. Classification is the last part of the block diagram in Figure 1 before updating the model parameters and classifying a sample. In the experiments of this study, classifiers depicted in Sections 3.3.1, 3.3.3 and 3.3.4 are trained using MATLAB and optimized using Bayesian optimization³, and the classifier depicted in Section 3.3.2 is trained using Python and optimized using the default parameter RMSprop.

3.3.1 Support Vector Machine (SVM)

A support vector machine (SVM) is a linear classifier for binary classification (Boser et al., 1992). The basic principle of an SVM is to find an optimal hyperplane that separates the data points of two classes and simultaneously maximizes the boundary between the two classes. For N data points

$$(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N) \in \mathcal{X} \times \{\pm 1\}, \quad i = (1, \dots, N), \quad (3.8)$$

³ For a detailed description and an overview, see e.g. (Tamura & Hukushima, 2018).

where x_i are observations, y_i are labels⁴, and χ is a set containing all observations x_i . With a weight vector \mathbf{w} , a hyperplane can be written in the form

$$\mathbf{w} \cdot \mathbf{x} - b = 0, \quad (3.9)$$

where $b \in \mathbb{R}$. By normalizing the dataset, i.e. rescaling \mathbf{w} and b so that the data points closest to the optimal hyperplane satisfy the equation

$$|\mathbf{w} \cdot \mathbf{x}_i + b| \geq 1, \quad (3.10)$$

we get the optimal hyperplane into the form

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1. \quad (3.11)$$

Equation (3.11) can be rewritten in the form

$$\mathbf{w} \cdot \mathbf{x}_i + b \geq 1 \text{ for } y_i = +1 \quad (3.12)$$

and

$$\mathbf{w} \cdot \mathbf{x}_i + b \leq -1 \text{ for } y_i = -1. \quad (3.13)$$

Now the optimal hyperplane can be constructed by solving the equation

$$\text{minimize } \tau(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2 \text{ subject to } y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1. \quad (3.14)$$

This can only be achieved if the two classes are linearly separable. For non-separable classes, the objective function $\tau(\mathbf{w})$ is replaced in Equation (3.14) by

$$\tau(\mathbf{w}, \xi) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^N \xi_i, \quad (3.15)$$

where $C > 0$ is a penalty parameter and ξ_i are slack variables, both contributing to penalizing misclassifications (Schölkopf et al., 2002, pp. 1–17).

Some binary classification problems are not separable by a simple hyperplane. In these cases, the class of kernels k can be used to map x into a linear space \mathcal{H} using a function Φ , i.e.

$$\Phi : \chi \rightarrow \mathcal{H}, \quad (3.16a)$$

$$x \mapsto \mathbf{x} := \Phi(x), \quad (3.16b)$$

$$k(x, x') = \Phi(x) \cdot \Phi(x'). \quad (3.16c)$$

⁴ For mathematical purposes, the two classes are labeled +1 and -1.

In other words, $k(x, x')$ maps the dot product into a higher-dimensional linear space in which a hyperplane separating the classes can be found (Schölkopf et al., 2002, pp. 25–35). An example of the use of a kernel function for finding a hyperplane to separate data points is demonstrated in Figure 9.

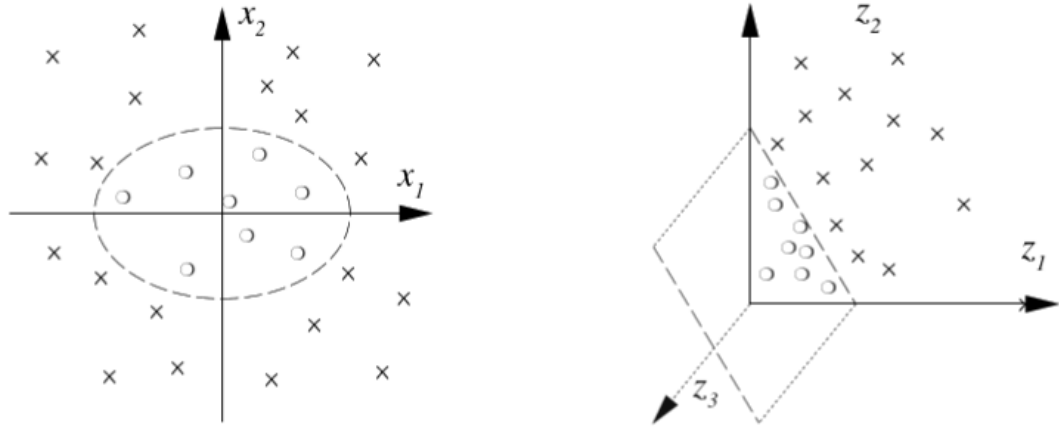


Figure 9. The use of a kernel function to find a separating hyperplane for data points (Schölkopf et al., 2002, p. 29).

In Figure 9, the left image depicts a situation where the data points marked with circles and crosses cannot be linearly separated from each other. As depicted in the right image, the use of a polynomial kernel function results in a new feature space where a hyperplane that perfectly separates the data points can be found. A few popular kernel functions used with SVMs are listed in Table 1.

Table 1. Popular kernel functions (Schölkopf et al., 2002, pp. 15–22, 115–118).

Kernel function name	Equation of $k(x, x')$
Gaussian or radial basis	$e^{-\ x-x'\ ^2}$
Linear	$x \cdot x'$
Polynomial of order d	$(x \cdot x')^d$

Hyperparameter optimization is a crucial part of finding an SVM suitable for a certain application (Rojas-Dominguez et al., 2018). There are multiple hyperparameters to be optimized, including the kernel function $k(x, x')$ used, the values of the weight vector w , the value of the constant b , the kernel scale parameter K , the penalty parameter C , and the slack variables ξ_i (Schölkopf et al., 2002, pp. 251–266, 281–309, 407–412). The hyperparameters are optimized based on the samples of the training set matrix X . The kernel scale parameter K divides all the values of X by K before applying the kernel

function $k(x, x')$. The parameters C and ξ_i prevent overfitting by imposing a penalty for misclassifications. If the prior probabilities of the elements in X are exceedingly unbalanced, an SVM can find a fit that favors one class over the other. To avoid this, the prior probabilities of the observations must be taken into account. The MATLAB implementation used in this study deals with this with the following algorithm:

1. Compute $p_c = \begin{bmatrix} p_1 \\ p_2 \end{bmatrix}$, where p_1 and p_2 are the prior probabilities of classes 1 and 2.
2. Remove observations from the training set X with zero probability.
3. Normalize weights w to sum up to the prior probability of the class which it belongs to, i.e. the weight for observation i in class k is

$$w_j^* = \frac{1}{\sum_{\forall i \in k} w_j} p_{c,k} .$$

Additionally, the penalty parameter C , also known as the box constraint, is assigned to every observation in X by MATLAB.

3.3.2 Multilayer Perceptron (MLP)

A multilayer perceptron (MLP) is a deep neural network (DNN) consisting of multiple smaller units called perceptrons (Rosenblatt, 1962). A perceptron is a binary classifier which determines its output using an activation function. A typical activation function is a logistic function (Chen et al., 2015)

$$f(x) = \frac{1}{1 + e^{-x}} , \quad (3.17)$$

but this study uses a rectifier function

$$f(x) = \max(0, x) , \quad (3.18)$$

also known as a rectified linear unit (ReLU), as it has been proven to yield better results compared to a logistic function because the error gradients backpropagate more efficiently through multiple layers (Glorot et al., 2011). This is because the derivative is always the same, independent of the value of the function, as long as it is positive. The use of a logistic function (3.17) might result in slow learning if the input values are large or very small, since in these cases the derivatives are close to zero which leads to the gradients being small.

The MLP network consists of an input layer, an output layer and an arbitrary number of hidden layers in between. These layers consist of nodes that are, apart from the input

nodes, neurons activated by an activation function. Typically, a neuron outputs 1 if it is activated by the input x , and 0 if activation does not occur. A suitable number of hidden layers and the number of neurons per each layer varies depending on the application (Rafiq et al., 2001).

After each iteration, every neuron updates its parameter values according to the output error of the network. Usually this is done by using gradient descent, an algorithm for finding the local minimum of a function by taking steps towards its negative of the gradient (Haykin, 1994). All connections in an MLP feed forward from one layer to the following layer without backward connections, and thus no connections form a loop. This results in the MLP being a feed-forward neural network (FNN), treating every input pattern independently without any memory over time (Schuller & Batliner, 2013, pp. 248–251). A neural network with a single hidden layer is capable of modeling any linear function, whereas a network with multiple hidden layers can perform non-linear classification (Minsky & Papert, 1988). The use of a dropout layer can help in avoiding overfitting when training a neural network. A dropout layer is a regularization technique in which the layer randomly deactivates a predetermined number of neurons at each iteration to prevent the system from overfitting. TensorFlow by Google (Zacccone et al., 2017) is a highly applicable and widely-used software used for training MLP networks, which is also used in this thesis.

3.3.3 Random Forests

A decision tree is a classification method in which the response to the given data is constructed from a tree-like model (Mitchell, 1997). The decision tree consists of a root node with N number of nodes, each node continuing a tree-like pattern of branching nodes performing binary decisions down to a leaf node containing the response. A depiction of a simple decision tree is demonstrated in Figure 10.

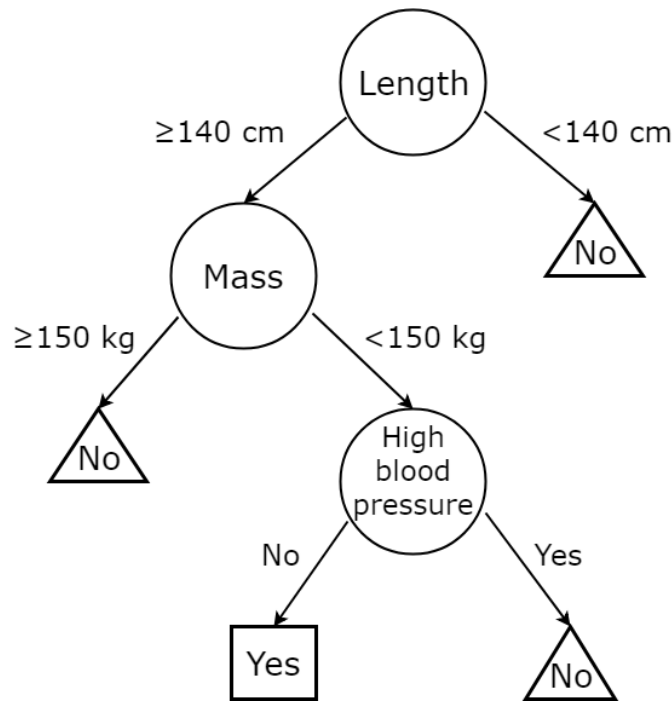


Figure 10. An example of a simple decision tree. The tree determines a Yes/No output whether a person is suitable for a roller coaster ride. If the test subject is too short, his/her mass exceeds limits or he/she has high blood pressure, then the person is not suitable for the ride. The leaf nodes are indicated with rectangles and triangles and nodes are indicated with circles, the root node being on top.

By combining a set of decision trees which are trained using randomly divided subsets of extracted features, we get a random forest. There are multiple ways of determining the output of an ensemble of trees (Rokach & Maimon, 2008, pp. 87–100), the simplest being the output of the majority of the trees in the forest. The MATLAB implementation used in the present experiments uses the empirical prior probabilities of the training set X and weights every observation with the prior probabilities in order to avoid favoring one observation over the other when predicting classes.

Hyperparameters that can be optimized with an ensemble of decision trees include the splitting decision values of each node, boosting, the randomization method, the number of learning cycles in the iteration process, shrinkage, the number of trees in the forest, and the minimum number of observations for every tree leaf (Breiman, 2001; Friedman, 2001; Rokach & Maimon, 2008, pp. 101–105). Boosting improves the performance of a weak learner by composing together classifiers produced by weak learners into a stronger classifier. There are multiple boosting algorithms, for example the AdaBoost algorithms AdaBoost.M1 and AdaBoost.M2 introduced by Freund and Schapire (1996). A typical randomization method used in random forests is bootstrap aggregating (Breiman, 2001), also known as bagging, in which the dataset is divided randomly into

subsets and the ensemble of decision trees is trained using the subsets. This leads to reducing the variance of the classifier and thus increases classification accuracy (Bühlmann & Yu, 2002).

The contributions of each tree in the ensemble can be slowed down using a weight ν . A shrinkage, or a learning rate, is the scaling of each updated value with ν (Friedman, 2001). Slowing down learning has been empirically proven to improve learning by reducing the influence of each tree, leaving room for the model to improve in future iterations (Friedman, 2002). As a trade-off, training with shrinkage increases the demand for more decision trees in the model, which results in increased computation time (Friedman, 2001).

3.3.4 k -Nearest Neighbors (k -NN)

Nearest neighbor (NN) classification algorithm (Cover & Hart, 1967) is based on comparing the feature vector of a given sample x in a test set to all the feature vectors in the training set and choosing the class of x based on the distance function $d(\mathbf{a}, \mathbf{b})$. The goal is to minimize $d(\mathbf{a}, \mathbf{b})$, i.e. to find the nearest neighbor x' for x , such that

$$x' = \arg \min_{\forall x_n \in X} d(x, x_n), \quad (3.19)$$

where X is the training set. There are many ways of determining the distance $d(\mathbf{a}, \mathbf{b})$ between two samples, and Table 2 shows a brief list of a few widely used distance functions.

Table 2. Distance functions used in NN classification (Hu et al., 2016; Lei, 2017, pp. 407–420).

Name	Equation of $d(\mathbf{a}, \mathbf{b})$	Additional notes
Euclidean distance	$\ \mathbf{a} - \mathbf{b}\ = \sqrt{\sum_{i=1}^M (a_i - b_i)^2}$	A special case of the Minkowski distance with $p = 2$.
Cosine distance	$1 - \frac{\mathbf{a}^T \mathbf{b}}{\ \mathbf{a}\ \cdot \ \mathbf{b}\ }$	
Minkowski distance	$\sqrt[p]{\sum_{i=1}^M a_i - b_i ^p}$	
City block distance	$\sum_{i=1}^M a_i - b_i $	A special case of the Minkowski distance with $p = 1$.
Chebyshev distance	$\max_i (a_i - b_i) = \lim_{p \rightarrow \infty} \sqrt[p]{\sum_{i=1}^M a_i - b_i ^p}$	A special case of the Minkowski distance with $p \rightarrow \infty$.

k -NN classification differs from NN classification so that, instead of determining the class of a sample x based on the class of its single nearest neighbor x' , x is classified based on the class of k of its nearest neighbors. The hyperparameters that can be optimized include the number of neighbors k , the distance function $d(\mathbf{a}, \mathbf{b})$ used, and the weight of the distance (Mullin & Sukthankar, 2000; Yang et al., 2006). The MATLAB implementation used in the present study compensates for the possible imbalance of the observations by weighting every observation with the prior probabilities of the observations in the training set.

4. EXPERIMENTAL SETUP

This chapter describes the experiments conducted in IDS/ADS classification using the supervised classification pipeline and methods described in the previous chapters. Additionally, this chapter discusses the dataset used in the project in detail.

4.1 Dataset

The given dataset, ACLEW starter set (Bergelson et al., 2017), consists of 5-minute-long recordings in MP3 format extracted from daylong recordings. The recordings were collected using a microphone worn by a child on a breast pocket, which means that the recordings are child-centered. The daylong recordings have not been modified in any way other than that the user-sensitive information has been cut out.

In the dataset, the daylong recordings were taken from 18 different families, six of them being from the United States and three from Canada, the United Kingdom, Mexico and Argentina. The US, UK and Canadian families speak English, the Argentinian families speak Spanish and the Mexican families consist of Tzeltal speakers with occasional Spanish. There is one 5-minute recording per each family in the dataset, totaling to 90 minutes of audio.

Individual utterances were extracted from the longer files using manual annotations of the utterances and their speaking style. The speaking styles were divided into four groups: IDS, ADS, speech directed to a composite group, and speech with the addressee undetermined. The two latter were left out from the utterance extraction, leaving only IDS and ADS samples. This resulted in the dataset being 1215 short audio samples with varying lengths, of which 850 ($\approx 70\%$) were IDS samples and 365 ($\approx 30\%$) were ADS samples. All of the samples were converted into WAV format before further processing. The lengths of the utterances varied widely since they ranged from 0.07 s to 14.86 s with an average of 1.42 s.

4.2 Setup

The features described in Section 3.2 were extracted from every utterance, and fixed-length feature vectors were obtained from the extracted features by taking the mean, variance, skewness and kurtosis of the time series of the data with the exception of STZCR, where the other functionals than the mean are not relevant. Different numbers of frequency bins in the range [32, 64, ..., 4096] were tested while taking the spectrogram

of every utterance with 128 bins performing the best. The features were normalized by scaling them to zero mean and unit standard deviation, after which the dataset was split into a training set and a test set in a ratio of 80:20 using random splitting. 25% of the test set was used as the development set for optimizing the hyperparameters.

SVM, k -NN, random forest and MLP classifiers were trained with 4-fold cross validation and optimized with different combinations of features with UAR (%), determined as

$$\text{UAR (\%)} = \frac{\frac{\text{correctly classified IDS samples}}{\text{IDS samples}} + \frac{\text{correctly classified ADS samples}}{\text{ADS samples}}}{2} \cdot 100\%, \quad (4.1)$$

as the primary evaluation measure of classification. By using UAR, models that take advantage of the imbalance of IDS/ADS samples in the test set are penalized. This is because labeling all output samples into IDS results in a classification accuracy of approximately 70% without the use of UAR, whereas UAR yields 50%. Bayesian optimization was used with SVMs, random forests and k -NN classifiers while RMSprop optimization was used with MLPs.

SVMs were trained with MATLAB with an iteration process where the hyperparameters were updated after every iteration. In the process, the values of the weight vector \mathbf{w} , the value of the constant b , the kernel scale parameter K , the box constraint C , and the slack variables ξ_i were optimized, as well as a suitable kernel function $k(x, x')$ was chosen. For moderate-dimensional data a polynomial kernel, and for higher-dimensional data a Gaussian kernel, yielded the best results. A standard MATLAB training routine was used for training in both k -NN and random forests with iterative training processes. For k -NN this translates into optimizing the number of neighbors k , the selection of the distance function $d(\mathbf{a}, \mathbf{b})$, and choosing the weight of the distance. Cosine distance and equal distance weight proved to be the most efficient for every feature vector, while the optimal number of neighbors varied depending on the length of the feature vector. Optimizing random forests involved optimizing the splitting decision values for each tree node, the learning method used, the number of nodes per each tree, the number of trees, the learning rate, and the minimum number of leaf node observations. These varied largely on the basis of the size of the feature vector, but with smaller dimensional features bagging, and with higher dimensional features AdaBoost.M1 were found to be the most efficient learning methods. MLPs were trained in Python with Keras using the TensorFlow backend with default parameters using binary cross-entropy as the loss function, and RMSprop as the optimizer except for using a ReLU (3.18) as the activation function. The best performance was achieved with a network consisting of two hidden layers with 1000 neurons each and a dropout layer in between, with a 50% random dropout.

After training SVM, k -NN, random forest and MLP classifiers with different feature vectors, the four best performing sets of features were taken into further examination, and they were used in building an ensemble classifier. These feature sets were MFCCs only (matrix size 1215x156), all features except spectrogram (1215x177), all features except MFCCs (1215x1045), and all features (1215x1201). The ensemble classifier was built by taking the vector of confidence values of each of the classifiers' predictions and adding them up into a combined confidence value vector. Then, a threshold for classifying a sample into IDS/ADS based on the confidence value was determined using the training set to obtain the best classification performance. By using confidence values instead of a majority voting scheme, a classifier with a certain decision contributes more to the final decision than classifiers with uncertain decisions. An ensemble classifier with a majority voting scheme was also tested, but it yielded far inferior results compared to using confidence values.

5. RESULTS

The classification results of the experimental setup are depicted in Table 3. The reported accuracies are calculated with the four best performing feature sets that were MFCCs only (matrix size 1215x156), all features except spectrogram (1215x177), all features except MFCCs (1215x1045), and all features (1215x1201).

Table 3. Training set and test set accuracies of the experimental setup with the four best performing sets of features: MFCCs only (x156), all features except spectrogram (x177), all features except MFCCs (x1045), and all features (x1201). Result accuracies are in UAR (%), where the best accuracy of a classifier is highlighted.

UAR (%)	Random forest		<i>k</i> -NN		SVM		MLP		Ensemble	
Features	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
x156	66.50	62.22	58.74	56.35	68.30	66.51	62.57	60.11	63.88	60.56
x177	59.22	56.65	62.91	58.80	67.89	66.25	68.02	64.85	70.42	69.20
x1045	58.34	56.06	55.55	54.49	59.53	59.08	58.47	56.35	61.27	59.91
x1201	61.72	59.19	57.35	56.26	60.76	60.34	61.00	59.97	63.14	62.01

Overall, the feature set with all extracted features except spectrogram (x177) performed the best, yielding the highest UAR (%) classification accuracy with the ensemble classifier. With the smallest feature set, random forests performed clearly the best, and SVMs performed slightly better than the second-best result of the same classifier. This can be explained by the fact that it is easier to find an optimal tree and forest structure as well as a separating hyperplane with smaller-dimensional data according to Occam's razor. The ensemble classifier gave the highest classification accuracy with 69.20% UAR, and in addition it gave the best accuracy compared to every other classifier except for the smallest feature set, where random forest and SVM classifiers performed better. The classification accuracies for the training sets were slightly higher than those for the test sets, indicating minor overfitting to the data but still being in tolerable limits.

6. CONCLUSION

The present study examined a system which classifies utterances into IDS and ADS as well as possible by experimenting with different classifiers and multiple sets of manually defined features. The results depicted in Chapter 5 were on par with the previous results in the fields of IDS/ADS classification and PSP with SVMs performing the best as individual classifiers, and the ensemble classifier outperforming all individual classifiers (Mahdhaoui et al., 2010; Schuster et al., 2014; Schuller et al., 2017). Of the feature sets used, MFCCs outperformed the spectrogram in terms of classification accuracies which indicates the significance of using features that correspond to the human perception of sound. MFCCs are lower dimensional and hence easier to model, and the use of MFCCs also partially removes irrelevant speaker-dependent variation such as the f_0 , and thus might work better. The spectrogram includes more speaker-dependent details, which results in the need of a larger dataset for a good classification model, whereas the MFCCs already attenuate some of the unnecessary details during feature extraction.

The classification results of this study can be considered as representative of classification performance expected in real-world use scenarios with unconstrained recording conditions. This is because the dataset used was realistic in terms of diversity, quality and the naturalness of the recordings. On the other hand, the utterances in the dataset are fully pre-segmented by hand, but in reality, the automatic segmentation of utterances is a substantially difficult task in complex recording environments. The random data division used in the study gives slightly optimistic results because some of the speakers in the training set and the test set were the same, and the corpus used is not very large. This is partially compensated by using cross-validation in the training of the models. The results are also marginally lower than they could be since the recordings used in the study are in MP3 format, while by using WAV files classification performance would improve slightly (as confirmed by in-lab experiments not reported here).

For further improving the classification system of this study, other functionals over time like the minimum, the maximum and the range should also be tested. Additionally, a larger pool of features extracted with, e.g., openSMILE, should be experimented with. Further examining which of the extracted features contribute to improving the classification accuracy could also be investigated. Classification accuracy could also be improved by enhancing the sound quality of the recordings by denoising the utterances in some way, since the recordings contained various natural noise sources like background noise

and clothes rubbing against the microphone. Also, the separation of speakers in the recordings would improve classification accuracy since the dataset contained many utterances with overlapping talkers.

As with PSP in general, the main challenge of creating the classifiers in this study was to avoid overfitting the models to the data. This was achieved in tolerable limits. To further improve the performance of the IDS/ADS classification system, a larger dataset would be required. Although this is a challenge due to the time-consuming manual annotation and the gathering of the data, some resources already exist for this purpose and should be utilized if the system would be deployed for actual use. Reliable systems capable of automatically segmenting, denoising and labeling are required to improve future research in the field of PSP.

REFERENCES

- Batliner, A., Schuller, B., Schaeffler, S. and Steidl, S. (2008). Mothers, adults, children, pets - towards the acoustics of intimacy. *In proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, Las Vegas, NV, USA, March 31 – April 4, pp. 4497–4500.
- Batliner, A., Steidl, S., Schuller, B., Seppi, D., Vogt, T., Wagner, J., Devillers, L., Vidrascu, L., Aharonson, V., Kessous, L. and Amir, N. (2010). Whodunnit – searching for the most important feature types signalling emotion-related user states in speech. *Computer Speech & Language*, 25(1), pp. 4–28.
- Bengio, Y., Courville, A. and Vincent, P. (2013). Representation learning: a review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8), pp. 1798–1828.
- Beranek, L.L., Mellow, T.J. and Ebrary, I. (2012). *Acoustics*. Oxford: Academic Press.
- Bergelson, E., Warlaumont, A., Cristia, A., Casillas, M., Rosenberg, C., Soderstrom, M., Rowland, C., Durrant, S. and Bunce, J. (2017). Starter-ACLEW. *Databrary*. Retrieved January 17, 2019 from <https://nyu.databrary.org/volume/390>.
- Boser, B., Guyon, I. and Vapnik, V. (1992). A training algorithm for optimal margin classifiers. *In proceedings of the fifth annual workshop on computational learning theory*, Pittsburgh, PV, USA, July 27–29, pp. 144–152.
- Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), pp. 5–32.
- Bühlmann, P. and Yu, B. (2002). Analyzing bagging. *The Annals of Statistics*, 30(4), pp. 927–961.
- Butler, S.C., O'Sullivan, L.P., Shah, B.L. and Berthier, N.E. (2014). Preference for infant-directed speech in preterm infants. *Infant Behavior and Development*, 37(4), pp. 505–511.
- Chen, Z., Cao, F. and Hu, J. (2015). Approximation by network operators with logistic activation functions. *Applied Mathematics and Computation*, 256, pp. 565–571.
- Cooper, R.P. and Aslin, R.N. (1990). Preference for infant-directed speech in the first month after birth. *Child development*, 61(5), pp. 1584–1595.
- Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1), pp. 21–27.
- Drugman, T. and Alwan, A. (2011). Joint robust voicing detection and pitch estimation based on residual harmonics. *In proceedings of INTERSPEECH 2011, 12th Annual Conference of the International Speech Communication Association*, Florence, Italy, August 27–31, pp. 1973–1976.
- Eaves, B.S., Feldman, N.H., Griffiths, T.L. and Shafto, P. (2016). Infant-directed speech is consistent with teaching. *Psychological review*, 123(6), pp. 758–771.

- Eyben, F., Weninger, F., Gross, F. and Schuller, B. (2013). Recent developments in openSMILE, the Munich open-source multimedia feature extractor. *In proceedings of the 21st ACM international conference on multimedia*, Barcelona, Spain, October 21–25, pp. 835–838.
- Fernald, A. (1985). Four-month-old infants prefer to listen to motherese. *Infant Behavior and Development*, 8(2), pp. 181–195.
- Fernald, A., Taeschner, T., Dunn, J., Papousek, M., De Boysson-Bardies, B. and Fukui, I. (1989). A cross-language study of prosodic modifications in mothers' and fathers' speech to preverbal infants. *Journal of child language*, 16(3), pp. 477–501.
- Freund, Y. and Schapire, R.E. (1996). Experiments with a new boosting algorithm. *In proceedings of the 13th International Conference on Machine Learning*, Bari, Italy, July 03–06, pp. 325–332.
- Friedman, J.H. (2001). Greedy function approximation: a gradient boosting machine. *The Annals of Statistics*, 29(5), pp. 1189–1232.
- Friedman, J.H. (2002). Stochastic gradient boosting. *Computational Statistics and Data Analysis*, 38(4), pp. 367–378.
- Furui, S. (1986). Speaker-independent isolated word recognition based on emphasized spectral dynamics. *In proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, Tokyo, Japan, April 7–11, pp. 1991–1994.
- Glorot, X., Bordes, A. and Bengio, Y. (2011). Deep sparse rectifier neural networks. *Journal of Machine Learning Research*, 15, pp. 315–323.
- Gold, B., Morgan, N. and Ellis, D. (2011). *Speech and audio signal processing: Processing and perception of speech and music*. 2nd ed. New York: Wiley.
- Haykin, S. (1994). *Neural networks: a comprehensive foundation*. New York: Macmillan College Publishing Company.
- Hu, L., Huang, M., Ke, S. and Tsai, C. (2016). The distance function effect on k-nearest neighbor classification for medical datasets. *SpringerPlus*, 5(1), pp. 1–9.
- Jay, F. and Institute of Electrical and Electronics Engineers (1984). *IEEE standard dictionary of electrical and electronics terms*. 3rd ed. New York: Institute of Electrical and Electronics Engineers.
- Kay, S.M. (1988). *Modern spectral estimation: Theory and application*. Englewood Cliffs, NJ: Prentice Hall.
- Lei, B. (2017). *Classification, parameter estimation, and state estimation: an engineering approach using MATLAB*. 2nd ed. Hoboken, New Jersey: Wiley.
- Mahdhaoui, A. and Chetouani, M. (2009). A new approach for motherese detection using a semi-supervised algorithm. *IEEE International Workshop on Machine Learning for Signal Processing*, pp. 1–6.
- Mahdhaoui, A., Chetouani, M. and Kessous, L. (2010). *Advances in nonlinear speech processing: Time-frequency features extraction for infant directed speech discrimination*, pp. 120–127. New York: Springer Berlin Heidelberg.

McMurray, B., Kovack-Lesh, K.A., Goodwin, D. and McEchron, W. (2013). Infant directed speech and the development of speech perception: Enhancing development or an unintended consequence? *Cognition*, 129(2), pp. 362–378.

Minsky, M. and Papert, S. (1988). *Perceptrons: an introduction to computational geometry*. 3rd pr. ed. Cambridge, MA: MIT Press.

Mitchell, T.M. (1997). *Machine learning*. New York: McGraw-Hill.

Mullin, M. and Sukthankar, R. (2000). Complete cross-validation for nearest neighbor classifiers. *In proceedings of the 17th International Conference on Machine Learning*, Stanford, CA, USA, June 29 – July 02, pp. 639–646.

Oppenheim, A.V. (1970). Speech spectrograms using the fast Fourier transform. *IEEE Spectrum*, 7(8), pp. 57–62.

Oppenheim, A.V., Schafer, R.W. and Buck, J.R. (1999). *Discrete-time signal processing*. 2nd ed. Upper Saddle River (NJ): Prentice-Hall.

O'Shaughnessy, D. (1999). *Speech communications: Human and machine*. 2nd ed. New York (NY): IEEE Press.

Pegg, J.E., Werker, J.F. and McLeod, P.J. (1992). Preference for infant-directed over adult-directed speech: Evidence from 7-week-old infants. *Infant Behavior and Development*, 15, pp. 325–345.

Pulkki, V. and Karjalainen, M. (2014). *Communication acoustics: an introduction to speech, audio and psychoacoustics*. 1st ed. Chichester, England: Wiley.

Rabiner, L.R. and Schafer, R.W. (2007). *Foundations and trends® in signal processing: Introduction to digital speech processing*, 1(1–2), pp. 1–194. Now Publishers Inc.

Rabiner, L.R. and Schafer, R.W. (2011). *Theory and applications of digital speech processing*. London; Upper Saddle River.

Rafiq, M.Y., Bugmann, G. and Easterbrook, D.J. (2001). Neural network design for engineering applications. *Computers and Structures*, 79(17), pp. 1541–1552.

Räsänen, O., Kakouros, S. and Soderstrom, M. (2018). Is infant-directed speech interesting because it is surprising? – Linking properties of IDS to statistical learning and attention at the prosodic level. *Cognition*, 178, pp. 193–206.

Rojas-Dominguez, A., Padierna, L.C., Carpio Valadez, J.M., Puga-Soberanes, H.J. and Fraire, H.J. (2018). Optimal hyper-parameter tuning of SVM classifiers with application to medical diagnosis. *IEEE Access*, 6, pp. 7164–7176.

Rokach, L. and Maimon, O. (2008). *Data mining with decision trees: Theory and applications*. Singapore: World Scientific.

Rosenblatt, F. (1962). *Principles of neurodynamics: Perceptrons and the theory of brain mechanisms*. United States: Springer Berlin Heidelberg.

Rowe, M.L. (2008). Child-directed speech: Relation to socioeconomic status, knowledge of child development and child vocabulary skill. *Journal of child language*, 35(1), pp. 185–205.

Saint-Georges, C., Chetouani, M., Cassel, R., Apicella, F., Mahdhaoui, A., Muratori, F., Laznik, M. and Cohen, D. (2013). Motherese in interaction: at the cross-road of emotion and cognition? (A systematic review). *PloS one*, 8(10), pp. e78103.

Schölkopf, B., Smola, A.J. and Ebrary, I. (2002). *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. Cambridge, Mass: MIT Press.

Schuller, B. and Batliner, A. (2013). *Computational paralinguistics: Emotion, affect and personality in speech and language processing*. 1st ed. Hoboken, New Jersey: John Wiley & Sons.

Schuller, B., Steidl, S., Batliner, A., Burkhardt, F., Devillers, L., Müller, C. and Narayanan, S. (2013). Paralinguistics in speech and language—state-of-the-art and the challenge. *Computer Speech & Language*, 27(1), pp. 4–39.

Schuller, B., Steidl, S., Batliner, A., Bergelson, E., Krajewski, J., Janott, C., Amatuni, A., Casillas, M., Seidl, A., Soderstrom, M., Warlaumont, A., Hidalgo Gadea, G., Schnieder, S., Heiser, C., Hohenhorst, W., Herzog, M., Schmitt, M., Qian, K., Zhang, Y. and Zafeiriou, S. (2017). The INTERSPEECH 2017 computational paralinguistics challenge: Addressee, cold & snoring. *In proceedings of INTERSPEECH 2018, 18th Annual Conference of the International Speech Communication Association*, Stockholm, Sweden, August 20–24, pp. 3442–3446.

Schuller, B., Weninger, F., Zhang, Y., Ringeval, F., Batliner, A., Steidl, S., Eyben, F., Marchi, E., Vinciarelli, A., Scherer, K., Chetouani, M. and Mortillaro, M. (2019). Affective and behavioural computing: Lessons learnt from the first computational paralinguistics challenge. *Computer Speech & Language*, 53, pp. 156–180.

Schuster, S., Pancoast, S., Ganjoo, M., Frank, M.C. and Jurafsky, D. (2014). Speaker-independent detection of child-directed speech. *IEEE Spoken Language Technology Workshop*, pp. 366–371.

Shami, M. and Verhelst, W. (2007). *Speaker classification II*; Automatic classification of expressiveness in speech: a multi-corpus study. 4441, pp. 43–56. Berlin: Springer Berlin Heidelberg.

Soderstrom, M. (2007). Beyond babytalk: Re-evaluating the nature and content of speech input to preverbal infants. *Developmental Review*, 27(4), pp. 501–532.

Spinelli, M., Fasolo, M. and Mesman, J. (2017). Does prosody make the difference? A meta-analysis on relations between prosodic aspects of infant-directed speech and infant outcomes. *Developmental Review*, 44, pp. 1–18.

Tamura, R. and Hukushima, K. (2018). Bayesian optimization for computationally extensive probability distributions. *PloS one*, 13(3), pp. e0193785.

Tohyama, M. and Koike, T. (1998). *Fundamentals of acoustic signal processing*. United States: Academic Press.

Yang, L., Jin, R., Sukthankar, R. and Liu, Y. (2006). An efficient algorithm for local distance metric learning. *In proceedings of the 21st national conference on Artificial intelligence - Volume 1*, Boston, MA, USA, July 16–20, pp. 543–548.

Zaccone, G., Karim, M.R. and Menshaw, A. (2017). *Deep learning with TensorFlow*. 1st ed. Great Britain: Packt Publishing.

APPENDIX A: MFCCS, DELTAS AND DELTA-DELTAS OF THE TEST SIGNAL

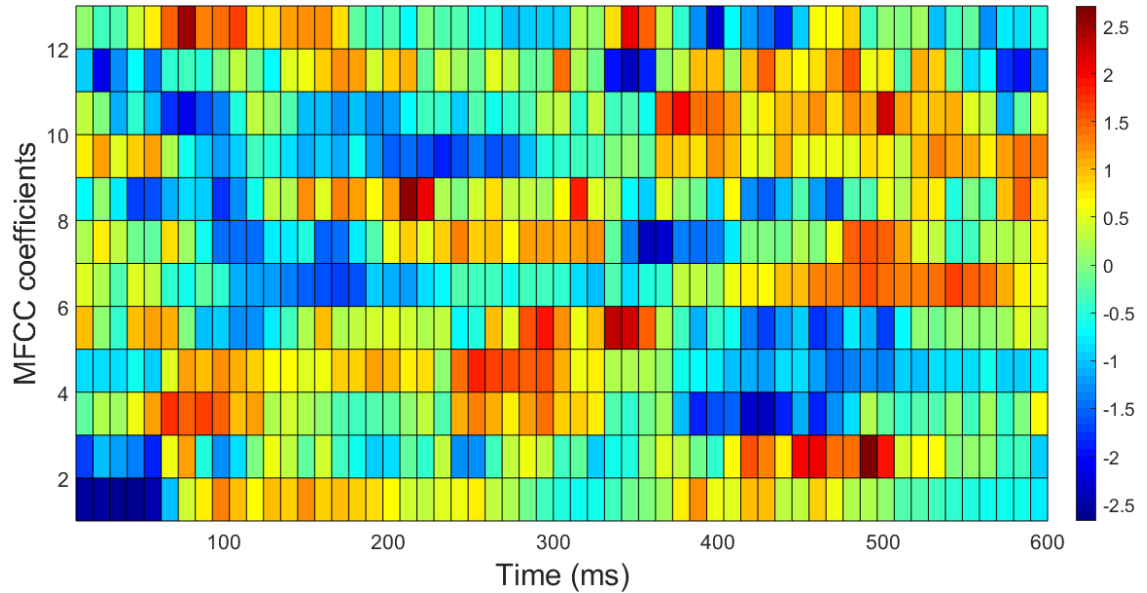


Figure 11. The MFCCs of the example signal in Figure 2 (z-scored for zero mean and unit variance).

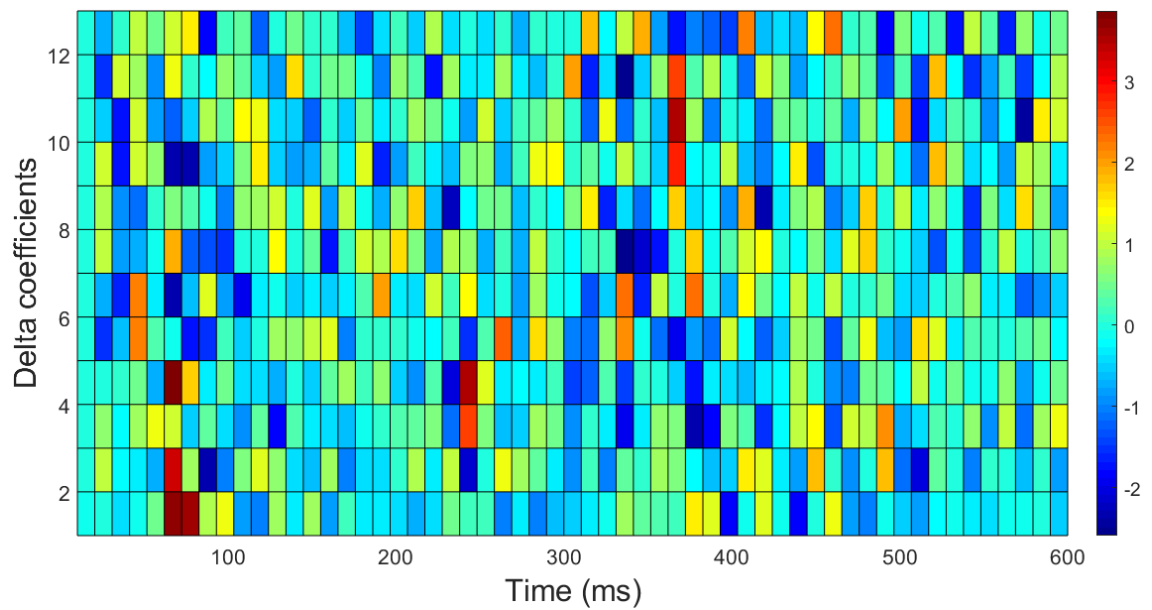


Figure 12. The deltas of the example signal in Figure 2 (z-scored for zero mean and unit variance).

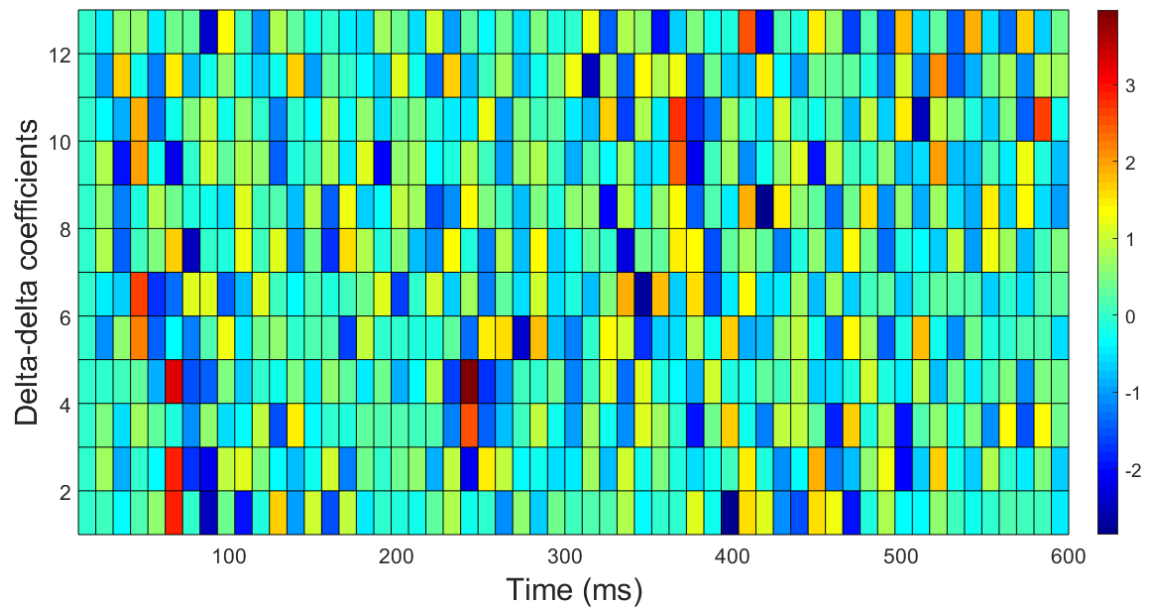


Figure 13. The delta-deltas of the example signal in Figure 2 (z-scored for zero mean and unit variance).